Rekodning i Stata med recode-kommando

Lolle, Henrik Lauridsen

Publication date:
2019

Rekodning i Stata med `recode`-kommando

Henrik Lauridsen Lolle, april 2019

1. Indledning

I det følgende gennemgås forskellige almindelige typer af rekodning i Stata med en række eksempler. Der ses alene på rekodninger ved brug af programmer fra en såkaldt "do file". Jeg anbefaler kraftigt, at man *ikke* foretager sine rekodninger via peg-og-klik alene, da man i så fald ikke har nogen logbog over, hvordan rekodningerne er foretaget. Det er sådan set ok at *generere* koden via peg-og-klik, hvorfra der kan kopieres og indsættes i do file, men det er min (og de fleste andres) vurdering, at det på sigt kan betale sig hurtigt at gå over til (stort set) udelukkende at foretage rekodninger ved også selv at skrive program-stumperne. Det skal pointeres, at teksten er ment som en introduktion og ikke en komplet gennemgang af rekodning i Stata. Når der i arbejdet med egne data efterfølgende opstår problemer, som man ikke kan finde svar på i denne tekst, henvises til Stata's egne hjælpefunktioner og manualer og til søgning på nettet. Der findes masser af gode hjemmesider[1], og ofte hjælper det at lave specifikke søgninger på det problem, man sidder med. Er man klar over, hvilken Stata-kommando man søger hjælp til, kan man som det første søge inde fra Stata-programmet. Den gennemgående kommando, der benyttes i denne tekst, er `recode`, og vil man finde ud af mere angående denne, kan man skrive `help recode` i Stata's kommando-linje og taste return, hvorefter der kommer en hjælpeside frem om kommandoen, og der er mulighed for at gå videre til en komplet manual-sektion i pdf-format om kommandoen.

En samlet do file, indeholdende samtlige kommandoer, der gennemgås i teksten, kan downloades fra on-line appendiks. Der benyttes kun datasæt med ganske få cases[2] og variabler, som indlæses direkte via `input`-kommandoen. Helt små data egner sig typisk godt til indlæring af rekodninger og beregninger af nye variabler.

Herunder indlæses numeriske data for fem variabler (id, holdning, holdn1, holdn2 og holdn3). Det skal tilføjes, at datasættet (lige som de øvrige i teksten) er fabrikeret til lejligheden, og det giver ingen mening ud over som eksempel på rekodning af variabler i forskellige situationer.

[1] Se fx https://stats.idre.ucla.edu/stata/

[2] Termerne case og enhed benyttes i flæng for det samme, og det vil i alle tilfælde i disse eksempler være lig med en (fabrikeret) respondent.

```
input id holdning holdn1 holdn2 holdn3
1    3    2    2    2
2    1    2    2    2
3    2    1    1    1
4    4    4    4    4
5    5    3    3    3
6    5    3    5    5
7    1    4    4    4
8    1    .    .a   8
9    2    3    3    3
10   5    5    5    5
11   3    .    .b   9
12   2    4    4    4
end
list
```

Så snart, man skriver `input` i starten af en programlinje, er Stata klar over, at det nu drejer sig om input at data, og at der efter `input`-kommandoen følger data for de variabler, man har angivet, samt at indlæsningen afsluttes med en `end`-kommando. Den vertikale forbindelsesstreg helt til venstre i billedet mellem `input` og `end` sætter Stata selv. Variablernes værdier er i eksemplet adskilt med tabuleringer, men jeg kunne også have nøjes med blot et enkelt blanktegn/mellemrum. Med en `list`-kommando i den sidste sætning af ovenstående program har jeg udskrevet hele datasættet, så det er nemmere at overskue, hvilke værdier der hører til hvilke variabler:

	id	holdning	holdn1	holdn2	holdn3
1.	1	3	2	2	2
2.	2	1	2	2	2
3.	3	2	1	1	1
4.	4	4	4	4	4
5.	5	5	3	3	3
6.	6	5	3	5	5
7.	7	1	4	4	4
8.	8	1	.	.a	8
9.	9	2	3	3	3
10.	10	5	5	5	5
11.	11	3	.	.b	9
12.	12	2	4	4	4

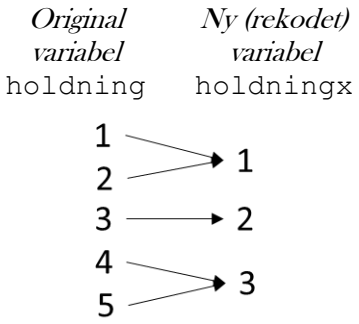
Det fremgår fx, at værdierne for de fem variabler i den første enhed er 1, 3, 2, 2 og 2. I case nr. 8 og 11 er der for nogle variabler angivet et punktum eller et punktum efterfulgt af et bogstav i stedet for en talkode. Det er koder for manglende data, og det kommer jeg nærmere ind på nedenfor. Den første variabel, `id`, er blot en fortløbende case-nummerering. Næsten alle datasæt har en variabel, der rummer et identifikationsnummer, og ofte vil enhederne være sorteret på det nummer. Ved nogle typer af analyse kan enhederne komme ud af den oprindelige sortering, men man kan altid ved hjælp af en `sort`-kommando vende tilbage til den oprindelige sortering.

2. Rekodning af ordinalskalerede variabler ved samling af kategorier

Efter id-variablen angives en variabel, der hedder `holdning`. Lad os sige, at den måler grad af enighed i et udsagn om, at den offentlige sektor er for stor, og at den måles på en skala fra

1 til 5, hvor 1 betyder "Meget uenig", og 5 betyder "Meget enig". Det fremgår af kolonnen af tal (anden kolonne), at der kun er tal mellem 1 og 5, dvs. ene valide værdier.

Der kan i mange situationer være gode argumenter for at rekode variabler til færre kategorier, dvs. samle nogle af kategorierne i grupper. I eksemplet her vil jeg rekode efter følgende retningslinjer:



Den ny variabel kalder jeg for `holdningx`, og jeg benytter kommandoen `recode` i Stata på følgende facon (efterfulgt af en frekvensudskrivning af både original og ny variabel):

```
recode holdning (1 2=1)(3=2)(4 5=3), gen(holdningx)
list holdning holdningx in 1/10
```

Den originale variabel angives umiddelbart efter kommando-navnet `recode`. Dernæst følger angivelse af rekodninger, og endelig – efter et komma – angiver jeg her, at den originale variabel ikke skal overskrives, men at der i stedet skal dannes en ny variabel med navnet `holdningx`. Det er vigtigt, at man husker det, med mindre at det er meningen, at den originale variabel skal gå tabt. Jeg har indsat en list-kommando efterfølgende, der udskriver værdier for case 1 til 10 for variablerne holdning og holdningx, blot for at vise resultatet af rekodningen, og det kan jo også benyttes som et hurtigt tjek for, hvordan rekodningen er forløbet, hvis man er i tvivl. Læg mærke til, hvordan en serie af cases kan beskrives som et interval, hvor start og slut er angivet med en skråstreg imellem. Se afsnit 4 for, hvordan man også kan benytte denne funktion i `recode`-kommandoen. Hvis man benytter list-kommandoen på datasæt med mange enheder, er det vigtigt at angive, hvilke cases og variabler, man vil have værdier for. I modsat fald bliver udskriften alt for voldsom. Jeg har et datasæt på kun 12 enheder, så i det følgende vil jeg udskrive værdier for samtlige enheder, når jeg benytter `list`-kommandoen. Resultatet af `list`-kommandoen er vist herunder, og det er heldigvis som forventet[3].

[3] Læg mærke til, at variabelnavnet `holdningx` er forkortet. Som default vises navnet kun med otte karakterer, og i "gamle dage" var det sådan set også kutyme (eller endog en regel), at variabelnavne i statistikprogrammer maksimalt skulle fylde otte karakterer. Man kan med inklusion af en option i `list`-kommandoen sætte et selvvalgt maksimalt antal karakterer. Skriv `help list` og return i kommandolinjen i Stata for at læse mere herom.

	holdning	holdni~x
1.	3	2
2.	1	1
3.	2	1
4.	4	3
5.	5	3
6.	5	3
7.	1	1
8.	1	1
9.	2	1
10.	5	3

Her i tekstens første rekodning gør jeg lige lidt ekstra ud af efterfølgende tjek, så jeg viser også herunder, hvordan de to variabler ser ud i en krydstabel. Først kommandoen og derefter resultatet:

```
tab holdning holdningx
```

	RECODE of holdning			
holdning	1	2	3	Total
1	3	0	0	3
2	3	0	0	3
3	0	2	0	2
4	0	0	1	1
5	0	0	3	3
Total	6	2	4	12

Efter variablen `holdning` følger i datasættet tre variabler med navnene `holdn1`, `holdn2` og `holdn3`. Det er tre versioner af samme holdningsvariabel, hvorfor de også har samme værdier/koder. Det er selvfølgelig urealistisk, at der i et og samme datasæt findes tre forskellige versioner af samme variabel, men for eksemplets skyld er det derimod fint. Det viser tre forskellige måder, man i Stata datasæt kan støde på ikke-valide værdier. Antag at der i det udsagn, der skulle svares på, også er givet mulighed for at svare "Ved ikke", og antag endvidere, at en del respondenter simpelthen har skullet springe over det spørgsmål som følge at et såkaldt filterspørgsmål forinden, hvorfor spørgsmålet kan siges at være irrelevant for de pågældende respondenter. "Ved ikke"-besvarelser kan selvfølgelig i nogle situationer betragtes som valide svar, der kan analyseres på, men i rigtig mange situationer vil man begrænse sig til at analysere på værdierne mellem 1 og 5, dvs. mere eller mindre enig i udsagnet.

Missing values er i Stata kodet som et punktum eller et punktum efterfulgt af et bogstav: ".", ".a", ".b", ".c" ... ".z". Punktum efterfulgt af et bogstav kan benyttes til at skelne mellem forskellige slags missing values, fx kunne "Ved ikke" tildeles koden ".a", mens "Irrelevant" kunne tildeles koden ".b". Men lad os til se på variablerne `holdn1` og `holdn2`. I variablen `holdn1` benyttes blot et punktum for missing values generelt, mens der er skelnet mellem to slags *missing values* i variablen `holdn2`. Jeg gentager fremgangsmåden fra den første

rekodning herunder, blot nu med to variabler på én gang. Man skal så blot huske at angive lige så mange nye variabler som originale[4].

```
recode holdn1 holdn2 (1 2=1)(3=2)(4 5=3), gen(holdn1x holdn2x)
list holdn1 holdn1x holdn2 holdn2x
```

	holdn1	holdn1x	holdn2	holdn2x
1.	2	1	2	1
2.	2	1	2	1
3.	1	1	1	1
4.	4	3	4	3
5.	3	2	3	2
6.	3	2	5	3
7.	4	3	4	3
8.	.	.	.a	.a
9.	3	2	3	2
10.	5	3	5	3
11.	.	.	.b	.b
12.	4	3	4	3

Det fremgår, at disse rekodninger går godt, selvom jeg ikke specificerede, hvad der skulle ske med missing-værdierne. Det forholder sig nemlig sådan, at de værdier/koder, der ikke nævnes i recode-kommandoen, blot kopieres over i de genererede variabler.

Den følgende variabel i datasættet, `holdn3`, er lidt anderledes mht. *missing values*. Det fremgår af udskriften af datasættet i indledningen ovenfor, men herunder viser jeg de tre variabler `holdn1`, `holdn2` og `holdn3` igen.

	holdn1	holdn2	holdn3
1.	2	2	2
2.	2	2	2
3.	1	1	1
4.	4	4	4
5.	3	3	3
6.	3	5	5
7.	4	4	4
8.	.	.a	8
9.	3	3	3
10.	5	5	5
11.	.	.b	9
12.	4	4	4

De svar, der i variablerne `holdn1` og `holdn2` er kodet som *missing values* (på forskellig vis), er i variablen `holdn3` i stedet for kodet med de talkoderne 8 og 9. Man vil ofte støde på Stata datasæt, hvor det ser sådan ud eller noget tilsvarende, fx også at "Ved ikke" og

[4] Hvis man har mange variabler, der skal rekodes på samme facon, kan man med fordel benytte prefix-funktionen, som gennemgås kortfattet i afsnit 6.

"Irrelevant" har negative værdier. Hvis jeg nu foretager rekodningen på samme måde som med `holdn1` og `holdn2`, kopieres koderne 8 og 9 blot over i den genererede variabel:

```
recode holdn3 (1 2=1)(3=2)(4 5=3), gen(holdn3x)
list holdn3 holdn3x
```

	holdn3	holdn3x
1.	2	1
2.	2	1
3.	1	1
4.	4	3
5.	3	2
6.	5	3
7.	4	3
8.	8	8
9.	3	2
10.	5	3
11.	9	9
12.	4	3

Det er selvfølgelig muligt, at det er sådan, man gerne vil have det, men det er også sandsynligt, at man hellere ville definere koderne 8 og 9 som missing. Hvordan man kan gøre det, viser jeg nedenfor på forskellig vis. Læg mærke til indsættelsen af tre skråstreger i tre af linjerne. Sådanne tre skråstreger i slutningen af en linje angiver, at kommandoen (Stata-sætningen) fortsættes på følgende linje. Pr. default vil et linjeskift nemlig i Stata betyde, at sætningen afsluttes. Den første recode-sætning i nedenstående program begynder med kommandoen `recode` og slutter altså i anden linje med `gen(holdn3xa)`.

```
recode holdn3 (1 2=1)(3=2)(4 5=3) ///
    (8 9=.), gen(holdn3xa)
recode holdn3 (1 2=1)(3=2)(4 5=3) ///
    (else=.), gen(holdn3xb)
recode holdn3 (1 2=1)(3=2)(4 5=3) ///
    (8=.a)(9=.b), gen(holdn3xc)
list holdn3 holdn3xa holdn3xb holdn3xc
```

	holdn3	holdn3xa	holdn3xb	holdn3xc
1.	2	1	1	1
2.	2	1	1	1
3.	1	1	1	1
4.	4	3	3	3
5.	3	2	2	2
6.	5	3	3	3
7.	4	3	3	3
8.	8	.	.	.a
9.	3	2	2	2
10.	5	3	3	3
11.	9	.	.	.b
12.	4	3	3	3

Der er næsten altid flere måder at gøre de samme ting på i Stata. I denne introduktionstekst viser jeg som regel blot en enkelt måde, men i dette eksempel giver det mening at vise lidt forskelligt. De to første kommandoer ovenfor udfører nøjagtigt det samme arbejde i den givne situation, nemlig at sørge for at koderne 8 og 9 bliver sat til *missing* i form af et punktum. Den første ved eksplicit at nævne de to koder, og den anden ved at bruge `else`-funktionen, som angår alle koder, der ikke er nævnt eksplicit i sætningen. At det giver samme resultat, skyldes, at der kun findes koderne 8 og 9, ud over dem, der allerede er nævnt (1 til 5) i recode-sætningen. Så vær opmærksom på, at else-funktionen drejer sig om alle andre koder, *missing* såvel som ikke *missing*. Så hvis der fx – ud over koderne 8 og 9 – også var *missing values* i form af fx et punktum, så ville disse blive kodet på samme vis i den nye variabel, altså som et punktum. Ud over `else`-funktionen, som altså angår alle ikke nævnte koder, kan man også angive `missing`, som angår alle ikke nævnte *missing*-koder, og man kan benytte `nonmissing`, som angår alle ikke nævnte *valide* koder.

Den tredje og sidste sætning i programmet gør det på en lidt anderledes facon, idet 8 og 9 kodes som forskellige typer *missing values*, henholdsvis ".a2 og ".b".

3. ANGIVELSE AF *VALUE LABELS* IFM. REKODNING

Denne tekst handler jo om rekodning i Stata, men fordi da angivelse af *value labels* kan integreres i `recode`-kommandoen, vil jeg kortfattet gennemgå denne funktion også. I afsnit 1 viste jeg, hvordan man kunne rekode en diskret variabel til færre kategorier. Uanset om der i den originale variabel var knyttet *values labels* til eller ej, vil de rekodede variabler ikke have *value labels*, fordi der alene er nævnt selve koderne/værdierne i de gennemgåede `recode`-sætninger.

I ovenstående lagde jeg en komplet udskrift af samtlige enheders værdier på en variabel til grund for en rekodning. Jeg kunne nemt på det lille datasæt se, at der var tale om værdierne 1 til 5 plus nogle missing values i variablerne `holdn1` til `holdn3`. Det er imidlertid noget mere problematisk at skaffe sig sådan et overblik i et datasæt på 1.000 enheder. Man kan selvfølgelig håbe på, at der foreligger en fejlfri kodebog, men det kan man bare ikke altid regne med, og det er under alle omstændigheder godt at tjekke lidt selv også. Hvad gør man så? Ja, så skriver man vel en frekvenstabel ud, og det har jeg gjort herunder for variable holdn3b.

```
tab1 holdn2
```

holdn2	Freq.	Percent	Cum.
1	1	10.00	10.00
2	2	20.00	30.00
3	2	20.00	50.00
4	3	30.00	80.00
5	2	20.00	100.00
Total	10	100.00	

Sådan en tabel giver imidlertid ikke noget rigtig godt overblik, da missing values ikke fremgår. I variablen findes jo også koderne ".a" og ".b". Man *kan* ganske vist få `tab`-kommandoen til at udskrive *missing values* også, men jeg vil her gerne reklamere for den brugerudviklede

kommando `fre`, som er super, når man har brug for overblik over *missing values*, *value labels* og koder. Den skal installeres først, og det gøres ganske simpelt ved at skrive `css install fre` i kommandolinjen og taste *return*. Når den én gang er installeret, ligger den klar som alle andre integrerede Stata-kommandoer (indtil at man installerer en ny version af Stata). Jeg prøver at bruge den herunder.

```
fre holdn2
```

```
holdn2
------------------------------------------------------------------------------
                    |      Freq.     Percent       Valid        Cum.
--------------------+---------------------------------------------------------
Valid       1       |          1        8.33       10.00       10.00
            2       |          2       16.67       20.00       30.00
            3       |          2       16.67       20.00       50.00
            4       |          3       25.00       30.00       80.00
            5       |          2       16.67       20.00      100.00
            Total   |         10       83.33      100.00
Missing     .a      |          1        8.33
            .b      |          1        8.33
            Total   |          2       16.67
Total               |         12      100.00
------------------------------------------------------------------------------
```

Nu fremgår både totalfordelingen på alle kategorier samt fordelingen på valide cases, og det svarer i øvrigt til en SPSS frekvenstabel. Der er nu et bedre grundlag at rekode ud fra, dog stadigvæk helst med et spørgeskema at støtte sig til også og gerne endvidere en decideret kodebog eller lignende. Jeg dropper/sletter den tidligere dannede variabel `holdn2x` og foretager rekodningen på ny, blot nu med tilknyttede *value labels*. Labels/betegnelser skal blot inkluderes i anførselstegn efter hver rekodning[5].

```
drop holdn2x
recode holdn2 (1 2=1 "Uenig")(3=2 "Hverken eller")(4 5=3 "Enig") ///
    (.a=.a "Ved ikke")(.b=.b "Irrelevant"), gen(holdn2x)
fre holdn2x
```

```
holdn2x -- RECODE of holdn2
------------------------------------------------------------------------------
                            |      Freq.     Percent       Valid        Cum.
----------------------------+---------------------------------------------------
Valid       1  Uenig         |          3       25.00       30.00       30.00
            2  Hverken eller |          2       16.67       20.00       50.00
            3  Enig           |          5       41.67       50.00      100.00
            Total             |         10       83.33      100.00
Missing     .a Ved ikke       |          1        8.33
            .b Irrelevant     |          1        8.33
            Total             |          2       16.67
Total                         |         12      100.00
------------------------------------------------------------------------------
```

De fleste sekundære datasæt, man får fingre i og analyserer på, indeholder *value labels* på både *valid* og *missing values*, sådan som fx ovenstående. Og hvis fx variablen `holdn2x` er en original variabel, der skal rekodes til en dummy med 0 for "Uenig" og "Hverken eller" samt 1 for "Enig", og at *missing values* skal bevares, så kan man nøjes med at benytte

[5] Anførselstegnene er kun nødvendige, hvis der i value labels indgår blanktegn.

ovenstående frekvenstabel fra `fre`-kommandoen som basis for rekodningen[6]. Det vil ikke umiddelbart være tilfældet med en tilsvarende fra en `tab`-kommando. I den fremgår nemlig hverken *missing values* eller de bagvedliggende koder, men kun *value labels*. Som nævnt kan det lade sig gøre ved hjælp af tab-kommandoen også, men det er noget mere bøvlet efter min mening (og gennemgås ikke her). Ud fra frekvenstabellen fra `fre`-kommandoen ovenfor kan jeg rekode til dummy på følgende måde:

```
recode holdn2x (1 2=0 "Uenig")(3=1 "Enig") ///
    (.a=.a "Ved ikke")(.b=.b "Irrelevant"), gen(holdn2d)
fre holdn2d
```

```
holdn2d ── RECODE of holdn2x (RECODE of holdn2)
```

		Freq.	Percent	Valid	Cum.
Valid	0 Uenig	5	41.67	50.00	50.00
	1 Enig	5	41.67	50.00	100.00
	Total	10	83.33	100.00	
Missing	.a Ved ikke	1	8.33		
	.b Irrelevant	1	8.33		
	Total	2	16.67		
Total		12	100.00		

4. Rekodning af intervalskalerede variabler

Når man skal rekode intervalskalerede variabler eller tilnærmelsesvis intervalskalerede variabler ved at samle i kategorier/intervaller, foregår det i hovedtræk på samme måde som beskrevet i afsnittet ovenfor vedrørende ordinalskalerede variabler. Der er dog enkelte aspekter, der er lidt specielle, hvorfor dette gennemgås i særskilt afsnit.

Jeg starter med at smide de gamle data ud (med `clear all`), indlæse nogle nye og udskrive variablernes værdier:

```
clear all
input id kvinde alder indkomst lykke
1   0   45  356     7
2   1   18  400     6
3   1   22  250     0
4   0   51  170     9
5   1   67  400.1   2
6   1   38  350     5
7   0   30  175     10
8   1   75  250     8
9   0   66  300     .
10  0   25  .       9
11  1   59  560     4
12  0   .   405     8
end
list
```

[6] Vær dog opmærksom på, at `fre`-kommandoen pr. default kun udskriver de 20 laveste værdier og de 20 største plus missing values, med mindre at man specificerer, at der skal gøres andet. Det har naturligvis ingen betydning her, da der kun er fem valide koder i alt.

	id	kvinde	alder	indkomst	lykke
1.	1	0	45	356	7
2.	2	1	18	400	6
3.	3	1	22	250	0
4.	4	0	51	170	9
5.	5	1	67	400.1	2
6.	6	1	38	350	5
7.	7	0	30	175	10
8.	8	1	75	250	8
9.	9	0	66	300	.
10.	10	0	25	.	9
11.	11	1	59	560	4
12.	12	0	.	405	8

Jeg har nu et nyt datasæt, stadigvæk blot 12 enheder, men med andre variabler. Dels har jeg en dummyvariabel (`kvinde`), dels har jeg tre intervalskalerede eller tilnærmelsesvis intervalskalerede variabler. Variablerne `alder` og `lykke` er ret nemme at have med at gøre, og rekodning af disse foregår på samme måde som ved ordinalskalerede variabler. Det er karakteristisk for de to variabler, at de alene kan antage heltalsværdier. Variablen *lykke* kan fx antage heltal mellem 0 og 10. Hvis jeg vil rekode den variabel til tre kategorier, hvor værdierne 0-3 kodes til 1, 4-6 kodes til 2, og 7-10 kodes til 3, så vil jeg umiddelbart vælge at kode som følger.

```
recode lykke ///
    (0/3=1 "Ulykkelig") ///
    (4/6=2 "Hverken eller") ///
    (7/10=3 "Lykkelig") ///
    , gen(lykkex)
```

Den eneste forskel fra rekodningen af de ordinalskalerede holdningsvariabler ovenfor er, at jeg her gør brug af muligheden for at skrive et interval som to tal med en skråstreg imellem. Jeg kunne for så vidt også have benyttet den funktion ifm. rekodningen af holdningsvariablerne med skalaen fra 1 til 5, men der ville ikke rigtig være noget vundet med det, og det ville ikke være blevet mere overskueligt. Med en 0 til 10-skala begynder det imidlertid at blive en gevinst, og med variabler som alder i hele år og indkomst i tusinder af kroner bør man helt sikkert benytte intervalfunktionen. Der er kun én slags missing value i variablen `lykke`, så jeg behøver ikke bekymre mig om at specificere *missing values* i rekodningen. Disse bliver blot kopieret over i den nye variabel, `lykkex`, som missing.

Ved intervalskalerede variabler, der er kontinuerte eller noget, der ligner, er det ofte bekvemt at benytte endnu en funktion i `recode`-kommandoen. Variablen `indkomst`, der måler personlig indkomst i tusinder kroner, er sådan en variabel. Forestil dig, at jeg vil rekode indkomstvariablen til tre intervaller/kategorier efter følgende retningslinjer:

- Lav: [minimum; 200]
- Middel:]200; 400]
- Høj:]400; maksimum]

De kantede parenteser angiver, at første interval skal gå til og med 200, mens andet interval skal gå til og med 400. Personer med en personlig indkomst på *over* 400.000 kr. er altså placeret i den høje indkomstkategori. Læg mærke til i udskriften af variablernes værdier ovenfor, at nummer to case/person er angivet med en værdi i indkomstvariablen på 400, dvs. en indkomst på 400.000 kr. Vedkommende skal med i mellemgruppen. Derimod skal den femte person, som er angivet med værdien 400,1 i indkomstvariablen, med i kategorien "Høj". En rekodning efter de kriterier kan gøres som nedenfor, hvor der i øvrigt gøres brug af de to *keywords* `min` og `max`, som står for henholdsvis minimumsværdi og maksimumsværdi[7].

```
recode indkomst (min/200=1 "Lav")(200/400=2 "Middel") ///
    (400/max=3 "Høj"), gen(indkomstx)
list indkomst indkomstx
```

	indkomst	indkom~x
1.	356	Middel
2.	400	Middel
3.	250	Middel
4.	170	Lav
5.	400.1	Høj
6.	350	Middel
7.	175	Lav
8.	250	Middel
9.	300	Middel
10.	.	.
11.	560	Høj
12.	405	Høj

Af udskriften af værdier fremgår bl.a., at personen med værdien 400 i indkomst havner i "Middel", mens personen med værdien 400,1 i indkomst havner i kategorien "Høj". Kommandoen `recode` retter sig efter den instruks, der bliver givet først, og det er, at 400 skal i "Middel". Når der så i næste interval nævnes en startværdi på 400, accepterer Stata ikke det som en kontraordre, men alt, der er det mindste større end 400, bliver kategoriseret som "Høj"[8].

---

[7] Det kan her tilføjes, at der med `max` menes maksimum blandt valide værdier. Det forholder sig nemlig sådan, at *missing values*, altså et punktum eller et punktum efterfulgt af et bogstav, af Stata defineres som de absolut højeste værdier overhovedet, men når man i en `recode`-sætning benytter *keyword* `max`, angår det altså alene valide værdier. Se mere herom i senere note (eller slides).

[8] For så vidt kunne man lige så godt have lavet andet interval som `(min/400=2 "Middel")` og tredje interval som `("min/max=3 "Høj")`. Det ville have givet samme resultat.

Hvis jeg i stedet blot vælger at benytte samme fremgangsmåde som ved lykkevariablen, ville jeg kode som herunder. Der er imidlertid et potentielt problem i sådan en rekodning. Der er et "hul" mellem værdierne 200 og 200,1. Hvad nu hvis der var en person i datasættet, der havde en værdi på 200,05? Sådan en person ville falde uden for rekodningerne og ville blot blive kopieret over i den nye variabel med værdien 200,05. En rekodning som denne fordrer altså, at man har styr på de mulige værdier. Har man det, er den til gengæld fin.

```
recode indkomst (min/200=1 "Lav")(200.1/400=2 "Middel") ///
    (400.1/max=3 "Høj"), gen(indkomstx)
```

Men hvad nu hvis i stedet for skulle rekodes efter disse retningslinjer (læg mærke til, hvordan paranteserne vender anderledes nu)?

- Lav: [minimum; 200[
- Middel: [200; 400[
- Høj: [400; maksimum]

I så fald skal man nævne intervallerne i omvendt rækkefølge i rekodningen:

```
recode indkomst (400/max=3 "Høj")(200/400=2 "Middel") ///
    (min/200=1 "Lav"), gen(indkomstxb)
```

Nu kommer værdien 200 med i "Middel" og 400 med i "Høj"[9].

Nu blev det vist lidt mere kompliceret, end hvad der var meningen. Men hvis blot man tjekker sine rekodninger, når man er i tvivl, så er det som regel ikke så slemt endda. Det plejer at lykkes efter lidt *trail and error*.

5. Betinget rekodning

Jeg har hidtil i teksten set på rekodninger, der skulle følge samme retningslinjer for samtlige enheder i datasættet. Der kan imidlertid sagtens opstå situationer, hvor dette ikke er tilfældet. Hvis fx at jeg vil kode en aldersvariabel i år om til en dummy, der angiver, om man er gammel eller ikke gammel, kan det tænkes, at man vil kode forskelligt for kvinder og mænd. Det kunne fx tænkes, at man ville kode kvinder som gamle, hvis de var 70 år eller derover, mens man ville kode mænd som gamle, hvis de var 65 år eller derover (selvom det virker lidt diskriminerende). En sådan rekodning kan vi ikke klare i én sætning. Man kan gøre det i Stata på forskellige måder, og flere af disse måder kan være lige gode. Man danner sig efterhåndennogle rutiner, som man arbejder godt med. Den måde, som jeg umiddelbart ville vælge, er følgende, hvor jeg først danner den nye variabel som en kopi af den originale, og dernæst rekoder med to forskellige recode-sætninger, én for kvinder og en anden for mænd, og hvor jeg overskriver værdierne i den nydannede variabel.

```
generate gammel=alder
recode gammel(min/65=0)(65/max=1) if kvinde==0
recode gammel(min/70=0)(70/max=1) if kvinde==1
```

I generate-kommandoen kopierer jeg blot alderen over i variablen gammel. I den første `recode`-sætning rekoder jeg efter retningslinjerne for mænd, men kun hvis variablen

[9] Hvis man i stedet for ville rekode på samme måde som ved lykkevariablen, skulle man skrive som nedenfor.

```
recode indkomst (min/199.9=1 "Lav")(200/399.9=2 "Middel") ///
    (400/max=3 "Høj"), gen(indkomstxc)
```

En programmør vil sandsynligvis kalde den form for "uskøn", ud over at det jo ikke i alle tænkelige situationer vil være korrekt. Her skal man så dog også tænke på, at en programmør ofte laver programmer, der netop kan gælde i alle situationer og fremtidige data, mens en samfundsforsker og -studerende som oftest vil lave programmer til ét specifikt datasæt.

`kvinde` er lig med 0, dvs. hvis det er en mand. I den anden `recode`-sætning rekoder jeg efter retningslinjerne for kvinder, men kun hvis variablen `kvinde` er lig med 1, dvs. hvis det er en kvinde. Det er her meget vigtigt at skrive dobbelt-lighedstegn i betingelserne. Det betyder stadigvæk blot *lig med*, men for Stata benyttes et enkelt lighedstegn alene i forbindelse med tilskrivning af en værdi i `generate`- eller `replace`-kommandoerne, fx som i "`generate var2=var1+5`", hvor variablen `var2` tilskrives værdien af var1 + tallet 5 (se senere note eller slides). Læg dernæst også mære til, at jeg i ovenstående `recode`-sætningerne denne gang ikke inkluderer et komma efterfulgt af en generering af ny variabel. Den ny variabel er jo allerede dannet i første sætning, så jeg overskriver blot værdierne i variablen `gammel` i de to `recode`-sætninger.

6. Rekodning af flere variabler på én gang

Til slut vil jeg kort gennemgå, hvordan man nemt kan rekode flere variabler i ét hug. Jeg har ganske vist ovenfor i afsnit 2 vist et enkelt eksempel herpå, men jeg vil gennemgå det lidt nøjere her og med flere funktioner. For at det kan lade sig gøre, kræves der dog, at de alle skal rekodes efter samme retningslinjer. Jeg gennemgår et eksempel med tre variabler, `holdn1`, `holdn2` og `holdn3`. Jeg indlæser datasættet her:

```
clear all
input id holdn1 holdn2 holdn3
1   2   3   2
2   2   2   1
3   1   3   1
4   4   3   2
5   3   4   1
6   3   5   3
7   4   4   2
8   .a  5   1
9   3   3   1
10  3   4   3
11  .b  .b  .b
12  4   4   2
end
```

Det er tre ordinalskalerede variabler, der kan antage heltalsværdier fra 1 til 5, og der er to typer af missing values, ".a" for "Ved ikke" og ".b" for "Irrelevant". En enkelt respondent har svaret ved ikke til et enkelt spørgsmål, og for en anden respondent har disse holdningsspørgsmål ikke været relevante. Jeg vil nu gerne rekode disse tre variabler sådan, at 1 og 2 kodes som 1 (uenig), 3 kodes som 2 (hverken eller), og 3 og 4 kodes som 3 (enig). Helt simpelt kan jeg gøre det på følgende måde, hvor jeg blot nævner alle variablerne, der skal kodes på samme måde i `recode`-sætningen, og hvor jeg genererer tre nye variabler. *Missing values* nævnes ikke i rekodningen og kopieres derfor blot over, som de er, dvs. som enten ".a" eller ".b".

```
recode holdn1 holdn2 holdn3 ///
    (1 2=1)(3=2)(4 5=3) ///
    , gen(holdn1x holdn2x holdn3x)
```

Efter rekodningen kan der oprettes *value labels* ved en `label define`-kommando, hvorefter disse *value labels* knyttes til de rekodede variabler ved en `label values`-kommando:

```
label define holdnlab ///
    1 "Uenig" ///
    2 "Hverken eller" ///
    3 "Enig" ///
    .a "Ved ikke" ///
    .b "Irrelevant"
label values holdn1x-holdn3x holdnlab
```

Rekodningen og labeltilknytningen kan imidlertid optimeres ved benyttelse af to options i `recode`-sætningen, dels en præfix-option vedrørende navngivelse af de nye rekodede variabler, dels en label-option. Des flere variabler, der skal rekodes på én gang, des mere vinder man ved præfix-muligheden, og hvis variablerne følger umiddelbart efter hinanden i datasættet, som ved et spørgsmålsbatteri, kan man nøjes med at nævne den første og den sidste variabel med en bindestreg imellem, sådan som det vises herunder.

```
recode holdn1-holdn3 ///
    (1 2=1 "Uenig")(3=2 "Hverken eller")(4 5=3 "Enig") ///
    (.a=.a "Ved ikke")(.b=.b "Irrelevant") ///
    , prefix(x) label(labholdn)
fre xholdn1-xholdn3
```

I eksemplet har jeg noteret et "x" som præfix (forstavelse), og de nye rekodede variabler kommer derfor til at hedde `xholdn1`, `xholdn2` og `xholdn3`. Til slut udskriver jeg frekvenstabeller på de tre variabler.

xholdn1 ── RECODE of holdn1

		Freq.	Percent	Valid	Cum.
Valid	1 Uenig	3	25.00	30.00	30.00
	2 Hverken eller	4	33.33	40.00	70.00
	3 Enig	3	25.00	30.00	100.00
	Total	10	83.33	100.00	
Missing	.a Ved ikke	1	8.33		
	.b Irrelevant	1	8.33		
	Total	2	16.67		
Total		12	100.00		

xholdn2 ── RECODE of holdn2

		Freq.	Percent	Valid	Cum.
Valid	1 Uenig	1	8.33	9.09	9.09
	2 Hverken eller	4	33.33	36.36	45.45
	3 Enig	6	50.00	54.55	100.00
	Total	11	91.67	100.00	
Missing	.b Irrelevant	1	8.33		
Total		12	100.00		

xholdn3 ── RECODE of holdn3

		Freq.	Percent	Valid	Cum.
Valid	1 Uenig	9	75.00	81.82	81.82
	2 Hverken eller	2	16.67	18.18	100.00
	Total	11	91.67	100.00	
Missing	.b Irrelevant	1	8.33		
Total		12	100.00		

6. KVIK GUIDE

Simpel rekodning af ordinalskaleret variabel (s. 3):

```
recode holdning (1 2=1)(3=2)(4 5=3), gen(holdningx)
list holdning holdningx in 1/10
```

Eksempler på rekodning, hvor bestemte talkoder kodes som missing i den nye variabel (s. 6):

```
recode holdn3 (1 2=1)(3=2)(4 5=3) ///
    (8 9=.), gen(holdn3xa)
recode holdn3 (1 2=1)(3=2)(4 5=3) ///
    (else=.), gen(holdn3xb)
recode holdn3 (1 2=1)(3=2)(4 5=3) ///
    (8=.a)(9=.b), gen(holdn3xc)
```

Rekodning af ordinalskaleret variabel inklusiv angivelse af value labels (s. 8):

```
recode holdn2 (1 2=1 "Uenig")(3=2 "Hverken eller")(4 5=3 "Enig") ///
    (.a=.a "Ved ikke")(.b=.b "Irrelevant"), gen(holdn2x)
```

Rekodning af tilnærmelsesvis intervalskaleret variabel kun med heltalsværdier (s. 10):

```
recode lykke ///
    (0/3=1 "Ulykkelig") ///
    (4/6=2 "Hverken eller") ///
    (7/10=3 "Lykkelig") ///
    , gen(lykkex)
```

Rekodning af intervalskaleret variabel, der også har karakter af en kontinuert variabel (s. 11):

```
recode indkomst (min/200=1 "Lav")(200/400=2 "Middel") ///
    (400/max=3 "Høj"), gen(indkomstx)
```

Samme som ovenfor, men hvor tallene 200 og 400 havner i en anden kategori end ovenfor (s. 12):

```
recode indkomst (400/max=3 "Høj")(200/400=2 "Middel") ///
    (min/200=1 "Lav"), gen(indkomstxb)
```

Betinget rekodning (s. 12):

```
generate gammel=alder
recode gammel(min/65=0)(65/max=1) if kvinde==0
recode gammel(min/70=0)(70/max=1) if kvinde==1
```

Rekodning af flere variabler på én gang (s. 13):

```
recode holdn1 holdn2 holdn3 ///
    (1 2=1)(3=2)(4 5=3) ///
    , gen(holdn1x holdn2x holdn3x)
```

Tildeling af value labels (s. 14)

```
label define holdnlab ///
    1 "Uenig" ///
    2 "Hverken eller" ///
    3 "Enig" ///
    .a "Ved ikke" ///
    .b "Irrelevant"
label values holdn1x-holdn3x holdnlab
```

Samme som ovenfor, men med kort udgave af variabelliste (brug af bindestreg) samt med prefix-funktion og labeltildeling i recode-sætningen (s. 14):

```
recode holdn1-holdn3 ///
    (1 2=1 "Uenig")(3=2 "Hverken eller")(4 5=3 "Enig") ///
    (.a=.a "Ved ikke")(.b=.b "Irrelevant") ///
    , prefix(x) label(labholdn)
fre xholdn1-xholdn3
```

Nedenstående kan benyttes til oprettelse og tilknytning af value labels sammen med ovenstående (s. 14):

```
label define holdnlab ///
    1 "Uenig" ///
    2 "Hverken eller" ///
    3 "Enig" ///
    .a "Ved ikke" ///
    .b "Irrelevant"
label values xholdn1-xholdn3 holdnlab
```